

WHAT IS CLAIMED IS:

1. In a computer environment where devices are occasionally connected together, a method for automated transmission and execution of an executable file of interest
5 originating from a digital camera, upon the digital camera's connection to a cellular phone, the method comprising:

connecting the digital camera to a cellular phone capable of hosting the camera;

10 identifying at least one particular cellular phone that is connected to the camera, including determining communication information allowing communication between the camera and the particular cellular phone, and determining command information allowing the camera to invoke execution of a file of interest at the particular cellular phone;

based on said determined communication information, transmitting the executable file of interest from said camera to the particular cellular phone; and

15 based on said determined command information, invoking execution of the executable file of interest after it has been transmitted to the particular cellular phone.

2. The method of claim 1, wherein said executable file of interest comprises a driver file.

20 3. The method of claim 2, wherein said driver file, upon execution, controls operation of said camera.

25 4. The method of claim 1, wherein said executable file comprises a binary file having instructions capable of executing at said cellular phone.

5. The method of claim 1, wherein said executable file comprises an application program capable of executing at said cellular phone.

6. The method of claim 1, wherein said camera includes an add-in device capable of being hosted by said cellular phone.

7. The method of claim 6, wherein said camera comprises a digital camera and wherein said method further comprises:

upon execution of said executable file at said cellular phone, transferring image information from said digital camera to said cellular phone.

8. At the method of claim 7, further comprising:

after transferring said image information from said digital camera to said cellular phone, wirelessly transmitting said image information to a third device.

9. The method of claim 1, wherein said cellular phone includes a computing device capable of hosting other devices.

10. The method of claim 1, wherein said cellular phone includes wireless transmission capability for transferring information received from said camera to other devices.

11. The method of claim 1, wherein said camera and cellular phones are occasionally connected together.

12. The method of claim 1, wherein said camera and cellular phones are permanently connected together.

13. The method of claim 1, wherein said camera and cellular phones are connected together via a serial communication link.

14. The method of claim 13, wherein said serial communication link

comprises an RS-232 serial communication link.

15. The method of claim 1, wherein said camera and cellular phones are connected together via a USB (Universal Serial Bus) link.

5

16. The method of claim 1, wherein invocation of said identifying step occurs upon connecting said camera and cellular phones together.

17. The method of claim 1, wherein said identifying step includes:
10 probing the camera's environment for determining which devices, if any, the camera is attached to.

18. The method of claim 17, wherein said probing step includes:
 determining a default communication medium for probing for new devices.

19. The method of claim 18, wherein said default communication medium is specified initially by factory-preset information.

20. The method of claim 18, wherein said default communication medium is a selected one of a wireless and a wired communication medium.

21. The method of claim 20, wherein said default communication medium includes a serial (RS-232) and a USB (Universal Serial Bus) wired communication medium.

22. The method of claim 19, wherein said factory-preset information is stored in a registry of the camera.

23. The method of claim 19, wherein said factory-preset information includes a default communication rate and default handshake protocol for at least one potential

cellular phone.

24. The method of claim 17, wherein said probing step includes:
executing an initial sequence of handshake commands and comparing any
response received to a list of known responses for identifying a particular cellular phone.

25. The method of claim 17, wherein said probing step continues until all
known potential cellular phones have been enumerated.

26. The method of claim 1, wherein said identifying step includes:
updating a registry at said camera for indicating any connected cellular phone
that has been identified.

27. The method of claim 1, further comprising:
upon identifying at least one particular cellular phone, ensuring that a state of
TCP/IP communication is reached between said camera and the particular identified cellular
phone.

28. The method of claim 27, wherein said step of ensuring that a state of
TCP/IP communication is reached includes:
initiating a PPP (Point-to-Point Protocol) communication session between said
camera and cellular phones; and, thereafter
initiating a TCP/IP communication session between said camera and cellular
phones.

29. The method of claim 27, wherein said step of ensuring that a state of
TCP/IP communication is reached includes:
determining an IP (Internet Protocol) address for said cellular phone.

30. The method of claim 1, wherein said step of transmitting the executable file of interest includes:
opening the executable file of interest at the camera; and
streaming the opened executable file of interest from the camera to the cellular phone.

31. The method of claim 30, wherein said streaming step includes:
employing XML protocol for packaging said executable file of interest for delivery to the cellular phone.

32. The method of claim 30, wherein said step of transmitting further comprises:
returning to said camera a file handle permitting said camera to access said executable file of interest transmitted to said cellular phone.

33. The method of claim 31, wherein said file handle comprises a file handle that may be understood by said cellular phone for accessing a particular file of interest at said cellular phone.

34. The method of claim 1, wherein said executable file of interest comprises a byte-code program, and wherein said cellular phone includes capability for executing byte-code programs.

35. The method of claim 1, wherein said executable file of interest comprises a Java program, and wherein said cellular phone includes a Java Virtual Machine for executing Java programs.

36. The method of claim 1, wherein said step of invoking execution of the executable file of interest includes:

issuing a command from said camera to said cellular phone to begin execution at said cellular phone of said executable file of interest.

37. The method of claim 1, wherein said step of invoking execution of the executable file of interest includes:

triggering execution of said executable file indirectly at said cellular phone by instructing said cellular phone to restart itself.

38. The method of claim 1, further comprising:

placing said camera in a listening mode, after said camera has invoked execution of said executable file at said cellular phone.

39. The method of claim 38, wherein said camera awaits commands from said cellular phone, while said camera is in a listening mode.

40. The method of claim 39, wherein commands received at said camera from said cellular phone control operation of said camera.

41. A multi-device system providing automated loading and execution of a driver required for connected devices, the system comprising:

a camera that may be connected to a cellular phone that is capable of hosting the camera; and

a subsystem, incorporated in the camera, for automatically:

(i) identifying the cellular phone upon connection to the camera, said subsystem initiating communication between the two devices;

(ii) uploading the driver of interest from the camera to the cellular phone; and

(ii) transmitting at least one command from the camera that invokes execution of the driver of interest at the cellular phone, whereupon the driver executes at the

cellular phone for controlling operation of the camera.

42. The system of claim 41, wherein said driver comprises a binary file having instructions capable of executing at said cellular phone.

5

43. The system of claim 42, wherein said binary file comprises native machine instructions for execution by a processor at said cellular phone.

44. The system of claim 42, wherein said binary file comprises byte-code instructions for execution by an interpreter at said cellular phone.

10

45. The system of claim 44, wherein said binary file comprises a Java program, and wherein said cellular phone includes a Java Virtual Machine for executing Java programs.

15

46. The system of claim 44, wherein said driver includes:
instructions for unpacking other executable files for execution at said cellular phone.

20

47. The system of claim 41, wherein said camera comprises an add-in device capable of being hosted by said cellular phone.

48. The system of claim 47, wherein said camera comprises a digital camera device, and wherein said cellular phone comprises a handheld device capable of hosting said digital camera device.

25

49. The system of claim 48, wherein said handheld computing device functions to retrieve digital image information from said digital camera device and wirelessly transmit that information to another system.

50. The system of claim 48, wherein said handheld device is a selected one of a cellular phone device and a handheld computing device.

51. In a computer environment where devices are occasionally connected together, a method for automated transmission, execution, and manipulation of an executable file of interest originating from a first device, upon the first device's connection to a host device, the method comprising:

connecting the first device to at least one other device capable of hosting the first device;

identifying at least one particular host device that is connected to the first device, including determining communication information allowing communication between the first device and the particular host device, and determining command information allowing the first device to manipulate and invoke execution of an executable file of interest at the particular host device;

based on said determined communication information, transmitting the executable file of interest from said first device to the particular host device;

based on said determined command information, transmitting from said first device to the particular host device commands that manipulate the executable file of interest at the particular host device; and

initiating a dialog between the two devices, including:

(i) executing said commands transmitted to the host device on the host device, and

(ii) in response to said commands transmitted to the host device, returning a reply from the host device to the first device.

52. The method of claim 51, wherein said commands include a command to load the executable file of interest.

53. The method of claim 51, wherein said commands include a command to

start the executable file of interest.

54. The method of claim 51, wherein said commands include a command to end the executable file of interest.

5

55. The method of claim 51, wherein said commands include a command to activate the executable file of interest.

10

56. The method of claim 51, wherein said commands include a command to get the capabilities of the host device.

57. The method of claim 51, wherein said commands include a command to get a reference to the executable file of interest that is running on the host device.

58. The method of claim 51, wherein said executable file of interest comprises a Java program, and wherein said host device includes a Java Virtual Machine for executing Java programs.

59. The method of claim 51, wherein said executable file of interest comprises a byte-code program, and wherein said host device includes capability for executing byte-code programs.

60. The method of claim 51, further comprising:
placing said host device in a listening mode to receive commands from said first device.

25

61. The method of claim 51, further comprising:
after said first device has transmitted a command to said host device, placing said first device in a listening mode to receive a reply transmitted from said host device.

62. The method of claim 51, wherein said reply transmitted from the host device in response to said command from the first device includes status information.

5 63. The method of claim 62, wherein said status information includes error information indicating an execution state of a preceding command executed at the host device.

10 64. The method of claim 51, wherein transmission between the devices employs XML protocol.

65. The method of claim 51, further comprising:
returning to said first device a file handle permitting said first device to access said executable file of interest while it resides at said host device.

66. The method of claim 65, wherein said file handle comprises a file handle that may be understood by said host device for accessing a particular file of interest at said host device.

20 67. The method of claim 51, wherein said dialog includes:
issuing a load application command from said first device to said host device to receive said executable file of interest transmitted from the first device.

25 68. The method of claim 67, wherein the load application command is transmitted from the first device to the host device as an XML stream with a syntax of:

```
<LoadApp>  
  <name> {app} </name>  
  <bin>  
    <size> {value} </size>
```

(data)
</bin>
</LoadApp>

5 69. The method of claim 67, wherein the reply to the load application command is transmitted by the host device to the first device as an XML stream with a syntax of:

10 <LoadAppR>
 <status>(value)</status>
 <handle>(value)</handle>
 </LoadAppR>

15 70. The method of claim 51, wherein said dialog includes:
 issuing a release application command from said first device to said host device to be able to delete said executable file of interest.

20 71. The method of claim 70, wherein the release application command is transmitted from the first device to the host device as an XML stream with a syntax of:

 <ReleaseApp>
 <handle>(value)</handle>
 </ReleaseApp>

25 72. The method of claim 70, wherein a reply to the release application command is transmitted by the host device to the first device as an XML stream with a syntax of:

30 <ReleaseAppR>
 <status>(value)</status>
 </ReleaseAppR>

73. The method of claim 51, wherein said dialog includes:

issuing a start application command from said first device to said host device to begin execution at said host device of said executable file of interest.

74. The method of claim 73, wherein the start application command is transmitted from the first device to the host device as an XML stream with a syntax of:

```
<StartApp>  
  <handle>(value)</handle>  //Handle to application  
</StartApp>
```

75. The method of claim 73, wherein the reply to the start application command is transmitted by the host device to the first device as an XML stream with the following syntax:

```
<StartAppR>  
  <status>(value)</status>  //Standard error replies  
</StartAppR>
```

76. The method of claim 51, wherein said dialog includes:
issuing a stop application command from said first device to said host device to discontinue execution at said host device of said executable file of interest.

77. The method of claim 76, wherein the stop application command is transmitted from the first device to the host device as an XML stream with a syntax of:

```
<StopApp>  
  <handle>(value)</handle>  
  <priority>(value)</priority>  
</StopApp>
```

78. The method of claim 76, wherein the reply to the stop application

command is transmitted by the host device to the first device as an XML stream with a syntax of:

```
<StopAppR>  
    <status>(value)</status>  
</StopAppR>
```

79. The method of claim 51, wherein said dialog includes:
issuing an activate application command from said first device to said host
device to bring current execution of said executable file of interest to the forefront at said
host device.

80. The method of claim 79, wherein the activate application command is
transmitted from the first device to the host device as an XML stream with a syntax of:

```
<ActivateApp>  
    <handle>(value)</handle>  
    <priority>(value)</priority>  
</ActivateApp>
```

81. The method of claim 79, wherein a reply to the activate application
command is transmitted by the host device to the first device as an XML stream with a syntax
of:

```
<ActivateAppR>  
    <status>(value)</status>  
</ActivateAppR>
```

82. The method of claim 51, wherein said dialog includes:
issuing a command to get information about device capabilities of said host
device.

83. The method of claim 82, wherein the device capabilities command is transmitted from the first device to the host device as an XML stream with a syntax of:

5 <GetCap>
 </GetCap>

84. The method of claim 82, wherein the reply to the get capabilities command is transmitted by the host device to the first device as an XML stream with a syntax of:

10 <GetCapR>
 <status>(value)</status>
 <lang>(value)</lang>
 <id>(value)</id>
 <imei>(value)</imei>
 <imsi>(value)</imsi>
 <screen>(value)</screen>
 <version>(value)</version>
 <dataLink>(value)</dataLink>
 <flash>(value)</flash>
 <cpu>(value)</cpu>
 </GetCapR>

85. The method of claim 51, wherein said dialog includes:
25 issuing a command to get information about an active application handle for the executable file of interest.

86. The method of claim 85, wherein the command to get information about an active application handle is transmitted from the first device to the host device as an XML stream with a syntax of:

30 <GetActAppHandle>
 </GetActAppHandle>

87. The method of claim 85, wherein a reply to the command to get information about an active application handle is transmitted by the host device to the first device as an XML stream with a syntax of:

5 <GetActAppHandler>
 <status>(value)</status>
 <handle>(value)</handle>
 </GetActAppHandler>

10